Using Loops and Nested Loops in MATLAB

Kshitiz Mangal Bajracharya

2nd October 2020

What do we mean by loop?

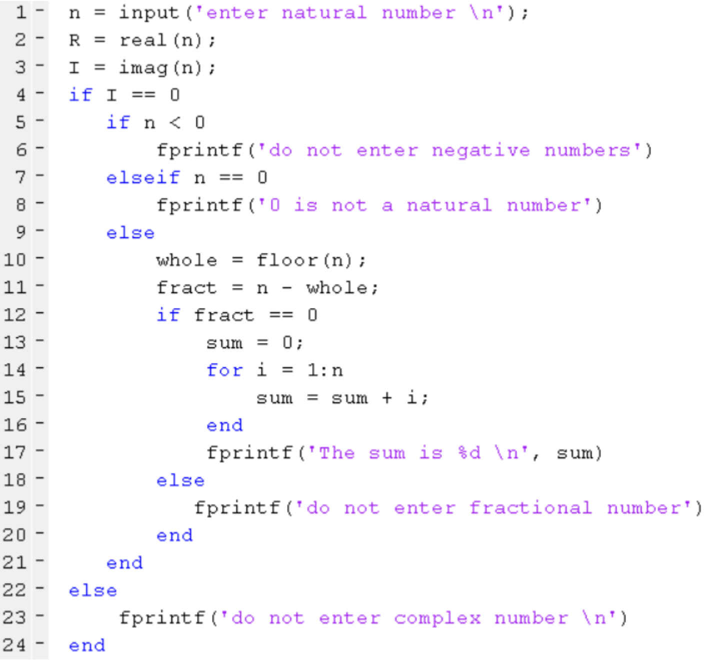
- A loop is a type of control command in MATLAB (and also other in other programming languages) which allows us to execute a block of codes more than once (in general), provided that the necessary conditions are satisfied.
- MATLAB uses two types of loops - `for` and `while`.
- A loop must be ended with an `end` command.
- If not written properly, a loop might be endless or it might not execute the necessary codes at all.

Using `for` loops

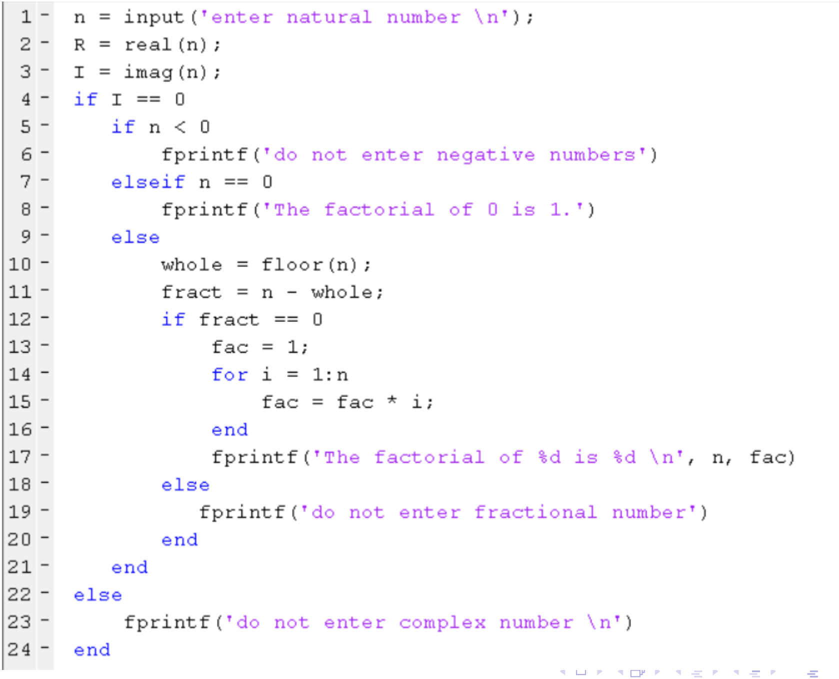
- The `for` loops are used when we are sure about the number of steps going to be involved in our program.
- The syntax of the `for` loop is given below.

```
for control_expression
      block of codes to be controlled by loop
end
```

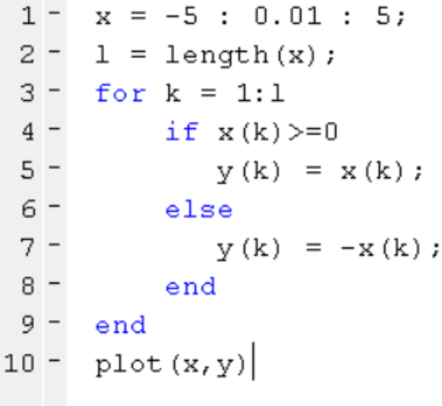
- The loop terminates only when the control expression (also called "loop index") is violated.
- If the control expression is violated at the beginning, the codes controlled by the loop will not executed at all.



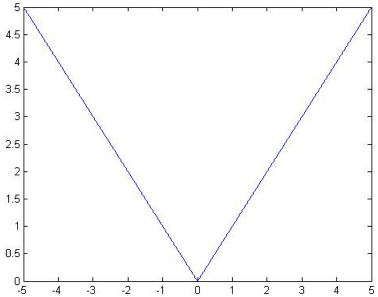
```
n = input('enter natural number \n');
R = real(n);
I = imag(n);
if I == 0
   if n < 0
       fprintf('do not enter negative numbers')
   elseif n == 0
       fprintf('0 is not a natural number')
   else
       whole = floor(n);
       fract = n - whole;
       if fract == 0
           sum = 0;
           for i = 1:n
               sum = sum + i;
           end
           fprintf('The sum is %d \n', sum)
       else
          fprintf('do not enter fractional number')
       end
   end
else
    fprintf('do not enter complex number \n')
end
```



```
n = input('enter natural number \n');
R = real(n);
I = imag(n);
if I == 0
   if n < 0
       fprintf('do not enter negative numbers')
   elseif n == 0
       fprintf('The factorial of 0 is 1.')
   else
       whole = floor(n);
       fract = n - whole;
       if fract == 0
           fac = 1;
           for i = 1:n
               fac = fac * i;
           end
           fprintf('The factorial of %d is %d \n', n, fac)
       else
          fprintf('do not enter fractional number')
       end
   end
else
    fprintf('do not enter complex number \n')
end
```



```
x = -5 : 0.01 : 5;
l = length(x);
for k = 1:l
    if x(k)>=0
        y(k) = x(k);
    else
        y(k) = -x(k);
    end
end
plot(x,y)
```

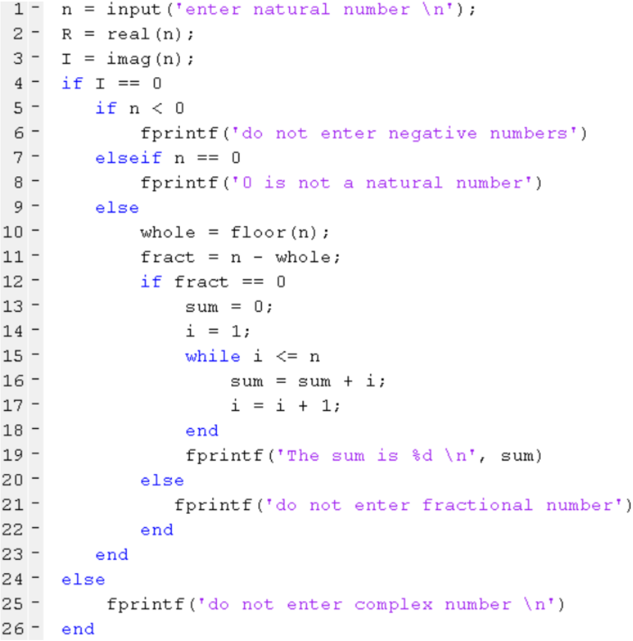


Using while loops

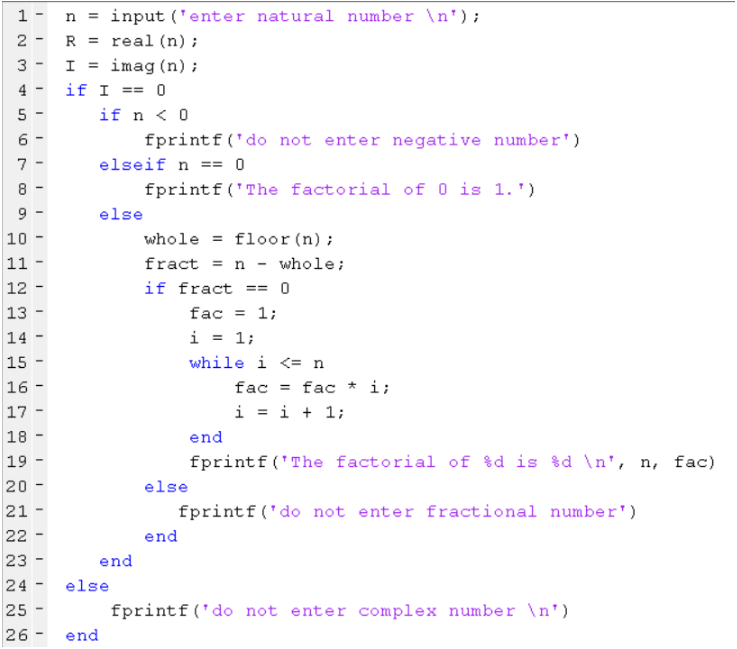
- The while loops are used when we are not sure about the number of steps going to be involved in our program.
- The syntax of the for loop is given below.

```
while control_expression
      block of codes to be controlled by loop
end
```

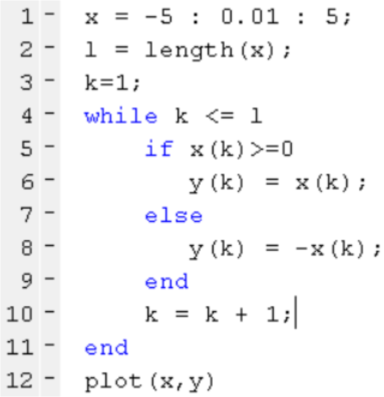
- The loop terminates only when the control expression (also called "loop index") is violated.
- If the control expression is violated at the beginning, the codes controlled by the loop will not executed at all.



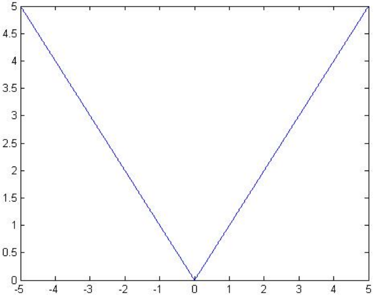
```matlab
n = input('enter natural number \n');
R = real(n);
I = imag(n);
if I == 0
   if n < 0
       fprintf('do not enter negative numbers')
   elseif n == 0
       fprintf('0 is not a natural number')
   else
       whole = floor(n);
       fract = n - whole;
       if fract == 0
           sum = 0;
           i = 1;
           while i <= n
               sum = sum + i;
               i = i + 1;
           end
           fprintf('The sum is %d \n', sum)
       else
          fprintf('do not enter fractional number')
       end
   end
else
    fprintf('do not enter complex number \n')
end
```



```
n = input('enter natural number \n');
R = real(n);
I = imag(n);
if I == 0
   if n < 0
       fprintf('do not enter negative number')
   elseif n == 0
       fprintf('The factorial of 0 is 1.')
   else
       whole = floor(n);
       fract = n - whole;
       if fract == 0
           fac = 1;
           i = 1;
           while i <= n
               fac = fac * i;
               i = i + 1;
           end
           fprintf('The factorial of %d is %d \n', n, fac)
       else
          fprintf('do not enter fractional number')
       end
   end
else
    fprintf('do not enter complex number \n')
end
```



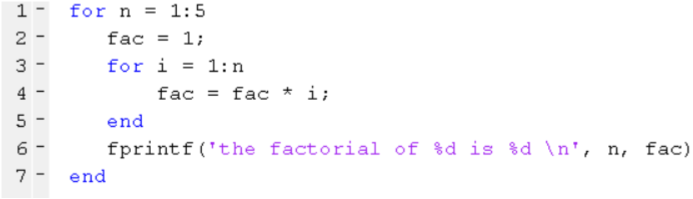
```
x = -5 : 0.01 : 5;
l = length(x);
k=1;
while k <= l
    if x(k)>=0
        y(k) = x(k);
    else
        y(k) = -x(k);
    end
    k = k + 1;
end
plot(x,y)
```



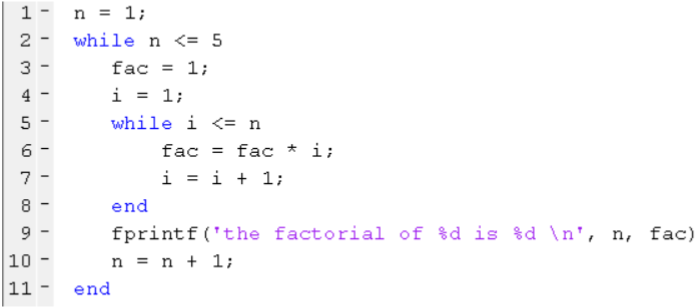
Nested loops

- If we use a loop (or multiple loops) within another loop, the resulting structure of MATLAB codes is called a nested loop.
- The basic nested loop structures in MATLAB can be
  - `for` within `for`
  - `while` within `while`
  - `for` within `while`
  - `while` within `for`

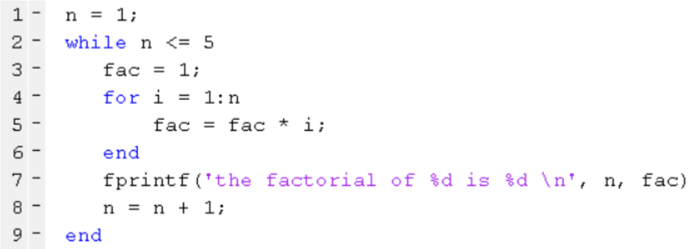
The respective examples of these type of nested loops are given below. All of them are related to displaying factorials of 1, 2, 3, 4 and 5.



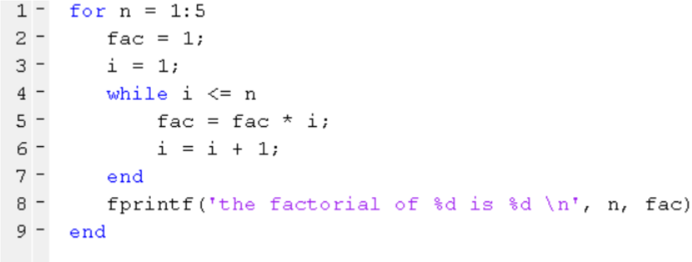
```
for n = 1:5
    fac = 1;
    for i = 1:n
        fac = fac * i;
    end
    fprintf('the factorial of %d is %d \n', n, fac)
end
```



```
n = 1;
while n <= 5
    fac = 1;
    i = 1;
    while i <= n
        fac = fac * i;
        i = i + 1;
    end
    fprintf('the factorial of %d is %d \n', n, fac)
    n = n + 1;
end
```



```
n = 1;
while n <= 5
    fac = 1;
    for i = 1:n
        fac = fac * i;
    end
    fprintf('the factorial of %d is %d \n', n, fac)
    n = n + 1;
end
```



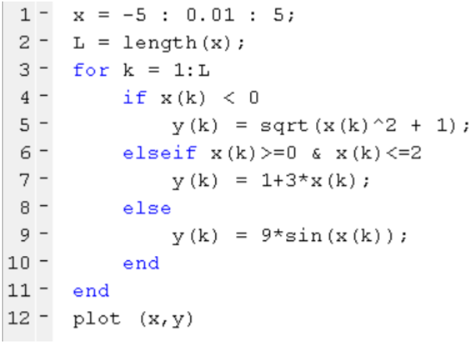
```
for n = 1:5
    fac = 1;
    i = 1;
    while i <= n
        fac = fac * i;
        i = i + 1;
    end
    fprintf('the factorial of %d is %d \n', n, fac)
end
```

Piece-wise defined functions

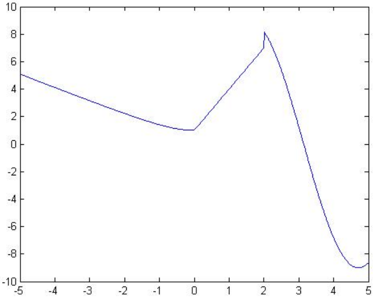
The following loop is for plotting the graph of the function $f : \mathbb{R} \to \mathbb{R}$ defined by

$$f(x) = \begin{cases} \sqrt{x^2+1} & x < 0 \\ 3x+1 & 0 \leq x \leq 2 \\ 9\sin x & x > 2 \end{cases}$$

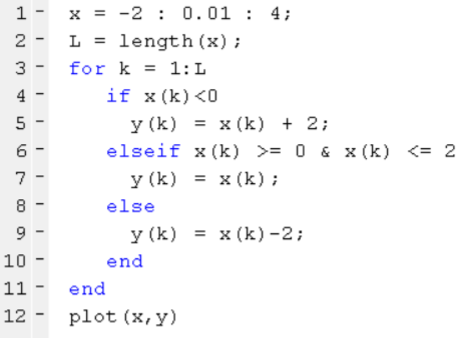
The MATLAB code is on the left and the output is on the right.



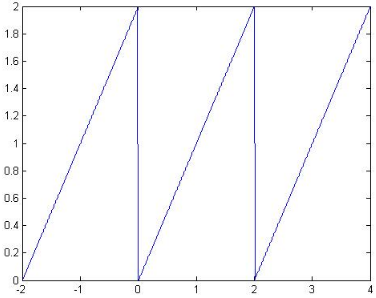
```
x = -5 : 0.01 : 5;
L = length(x);
for k = 1:L
    if x(k) < 0
        y(k) = sqrt(x(k)^2 + 1);
    elseif x(k)>=0 & x(k)<=2
        y(k) = 1+3*x(k);
    else
        y(k) = 9*sin(x(k));
    end
end
plot (x,y)
```



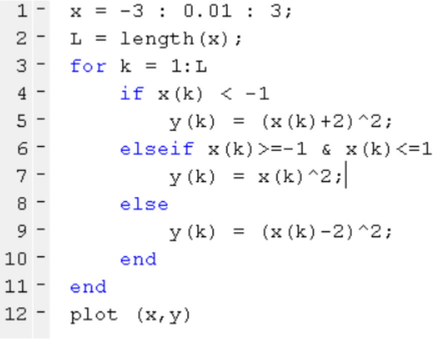
Periodic extension of $f(x) = x$ defined on $[0, 2]$ to the domain $[-2, 4]$.



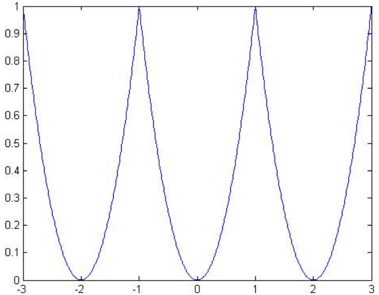
```
x = -2 : 0.01 : 4;
L = length(x);
for k = 1:L
   if x(k)<0
     y(k)  = x(k)  + 2;
   elseif x(k)  >= 0  & x(k)  <= 2
     y(k)  = x(k);
   else
     y(k)  = x(k)-2;
   end
end
plot(x,y)
```



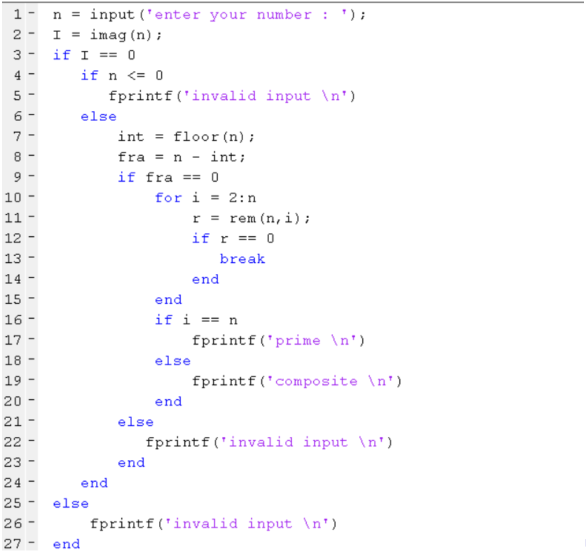
Periodic extension of $f(x) = x^2$ defined on $[-1, 1]$ to the domain $[-3, 3]$.



```
x = -3 : 0.01 : 3;
L = length(x);
for k = 1:L
    if x(k) < -1
        y(k) = (x(k)+2)^2;
    elseif x(k)>=-1 & x(k)<=1
        y(k) = x(k)^2;
    else
        y(k) = (x(k)-2)^2;
    end
end
plot (x,y)
```



Loop to test whether an input number is prime or composite



```
n = input('enter your number : ');
I = imag(n);
if I == 0
   if n <= 0
      fprintf('invalid input \n')
   else
       int = floor(n);
       fra = n - int;
       if fra == 0
           for i = 2:n
               r = rem(n,i);
               if r == 0
                  break
               end
           end
           if i == n
               fprintf('prime \n')
           else
               fprintf('composite \n')
           end
       else
          fprintf('invalid input \n')
       end
   end
else
    fprintf('invalid input \n')
end
```